Approximation Algorithms

PURPLE PETAL EDU HUB

Introduction

Approximation algorithms are algorithms used to find approximate solutions to optimization problems. Approximation algorithms are often associated with NP-hard problems; since it is unlikely that there can ever be efficient polynomial-time exact algorithms solving NP-hard problems, one settles for polynomial-time sub-optimal solutions. Approximation algorithms are increasingly being used for problems where exact polynomial-time algorithms are known but are too expensive due to the input size.

Objectives:

- To formalize the notion of approximation.
- To demonstrate several such algorithms.

Overview:

- Optimization and Approximation
- VERTEX-COVER, SET-COVER

Optimization

The task can be naturally rephrased as finding a maximal/minimal solution.

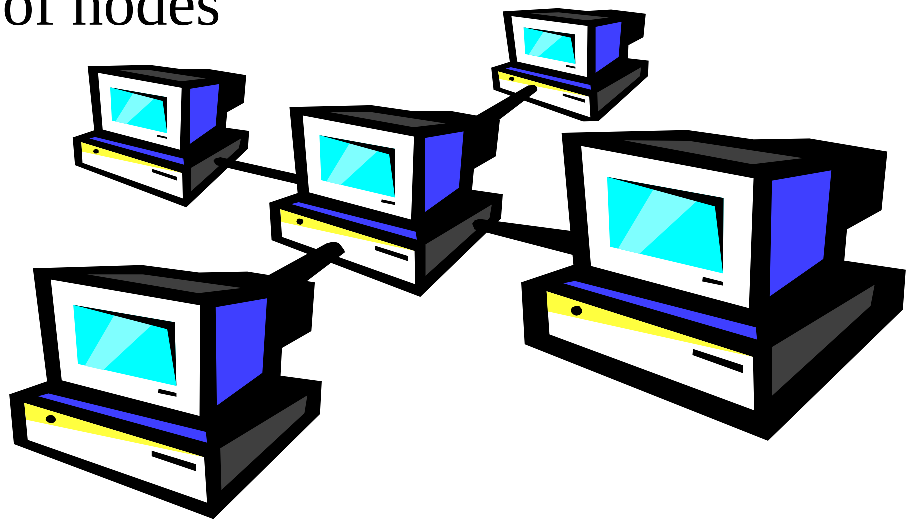
Approximation

An algorithm that returns an answer C which is "close" to the optimal solution C* is called an *approximation algorithm*.

Example:

Say you have a network, with links between some components

Each link requires power supply, hence, you need to supply power to a set of nodes that cover all links. Obviously, you'd like to connect the smallest number of nodes



Mass Mailing:

Say you'd like to send some message to a large list of people (e.g. all campus). You'd like to find the smallest set of lists that covers all recipients

Some Characteristics of Approximation Algorithms

- Time-efficient (sometimes not as efficient as heuristics)
- Don’t guarantee optimal solution
- Guarantee good solution within some factor of the optimum
- Rigorous mathematical analysis to prove the approximation guarantee
- Often use algorithms for related problems as subroutines

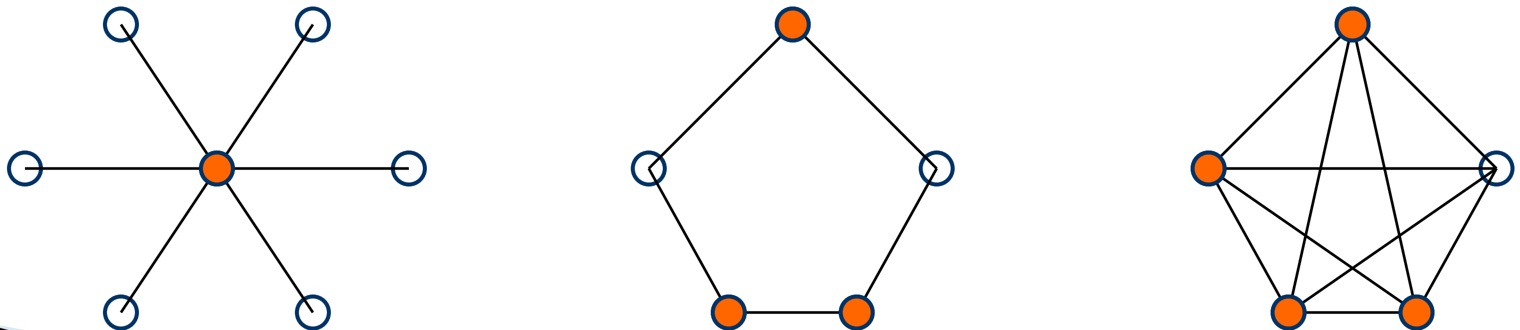
Vertex Cover

Vertex cover: a subset of vertices which "covers" every edge.

An edge is covered if one of its endpoint is chosen.

The Minimum Vertex Cover Problem:

Find a vertex cover with minimum number of vertices.



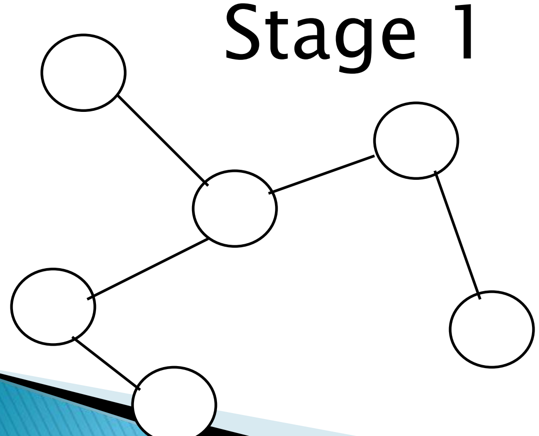
Travelling Salesman Problem

Given an instance for TSP problem,

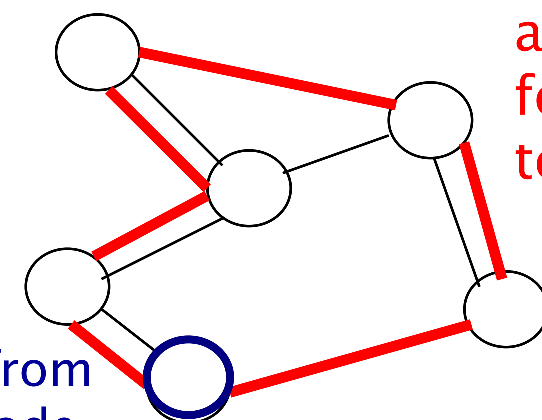
1. Find a minimum spanning tree (MST) for that instance.
2. To get a tour, start from any node and traverse the arcs of MST by taking shortcuts when necessary.

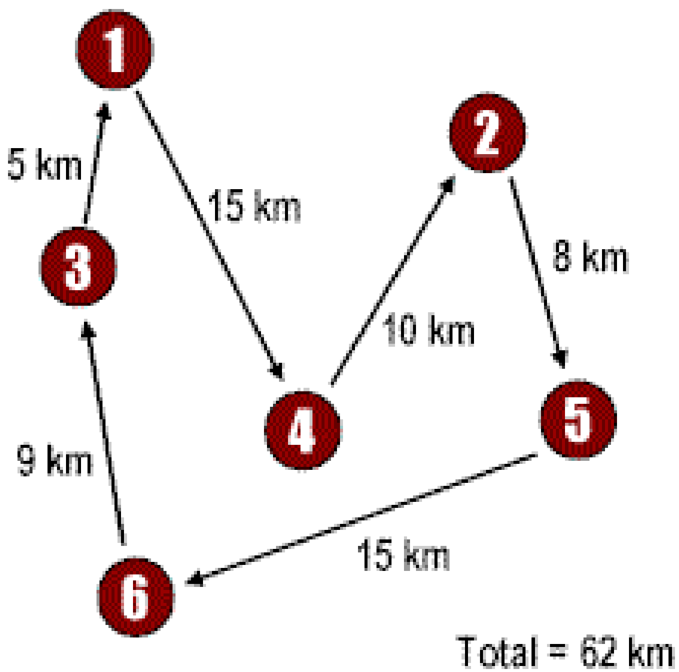
Example:

Stage 1

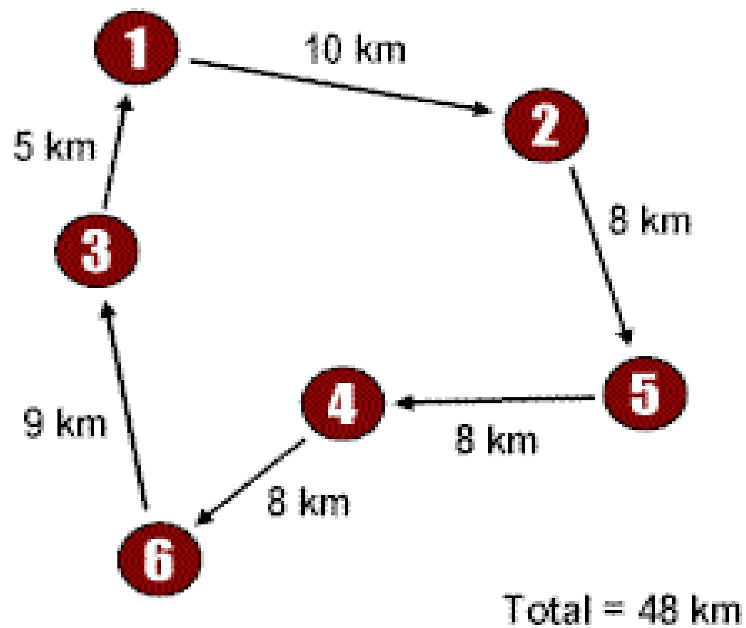


Stage 2





1
2
3
4
5
6
5 km
15 km
10 km
8 km
9 km
15 km
Total = 62 km



1
2
3
4
5
6
10 km
5 km
8 km
9 km
8 km
8 km
Total = 48 km

Absolute approximations

Planar graph coloring

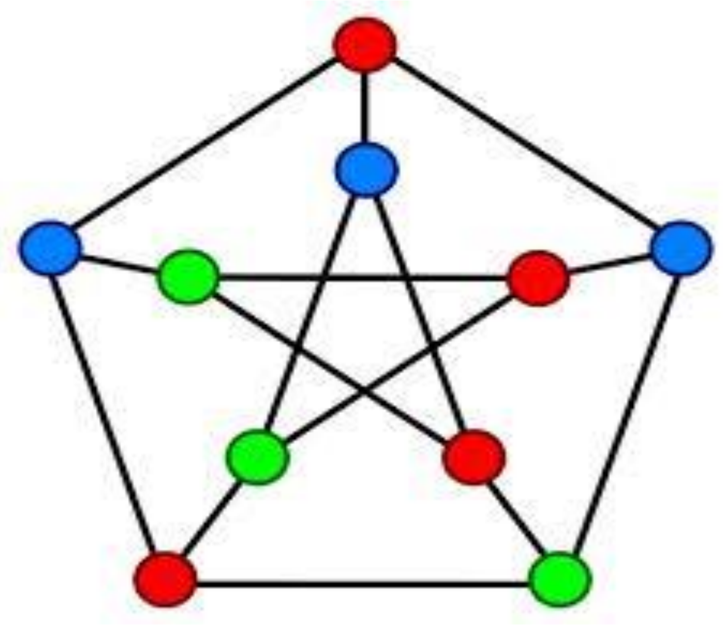
Assignment of colors (or labels) to vertices in a graph such that no two adjacent vertices share the same color

– Determine the minimum number of colors needed to color a planar graph G = (V;E)

– A graph is 0-colorable iff V =Ø ;

– A graph is 1-colorable iff E = Ø ;

– A graph is 2-colorable iff it is bipartite

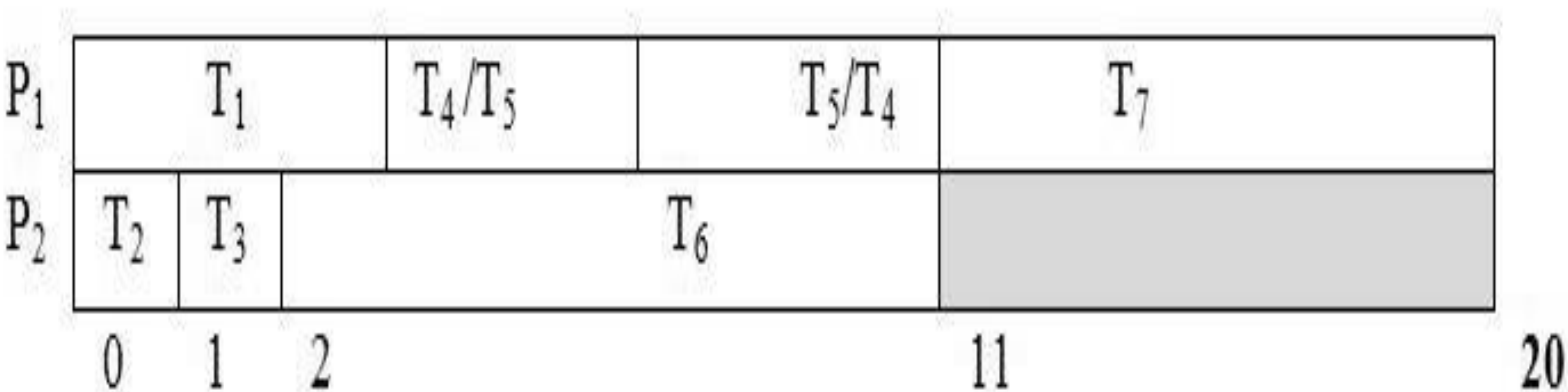


Scheduling independent tasks

Scheduling the n number of independent tasks such that m number of processors are assigned one process at a time.

T1	T2	T3	T4	T5	T6	T7
3	1	1	2	2	6	9

Tasks taking time t



Scheduling of tasks

Bin Packing Problem:

In the **bin packing problem**, objects of different volumes must be packed into a finite number of bins or containers each of volume V in a way that minimizes the number of bins used. In computational complexity theory, it is a combinatorial NP-hard When the number of bins is restricted to 1 and each item is characterised by both a volume and a value, the problem of maximising the value of items that can fit in the bin is known as the knapsack problem. Specifically, a set of items could occupy less space when packed together than the sum of their individual sizes.